数据库原理与应用设计

Principle，Application and Design of Database

主　　编：陶宏才

主　　审：尹治本

编写人员：陶宏才　楼新远

陈红梅　何　虎

陈安龙　苏金华

袁连海　吴　艾

桑应朋

西南交通大学出版社

·成　都·

图书在版编目（CIP）数据

数据库原理与应用设计/陶宏才主编. —成都 :西南交通大学出版社，2001.9（2003.6 **重印**）
ISBN 7-81057-581-3

Ⅰ. 数… Ⅱ. 陶… Ⅲ. ①关系数据库–数据库管理系统–理论②关系数据库–数据库管理系统–设计
Ⅳ. TP311. 13

中国版本图书馆 CIP **数据核字**（2001）**第** 063154 **号**

数 据 库 原 理 与 应 用 设 计

Principle, Application and Design of Database

主　编　陶宏才

主　审　尹治本

*

责任编辑　唐　晴
封面设计　肖　勤
西南交通大学出版社出版发行
（成都二环路北一段 111 **号　邮政编码：**610031　**发行部电话：**87600564**）**
http: //press.swjtu.edu.cn
E-mail: cbsxx@swjtu.edu.cn
四川森林印务有限责任公司印刷

*

开本：787 mm×1092 mm　1/16　**印张：**20.25
字数：430 **千字**　**印数：**4001—7000 **册**
2001 **年** 9 **月第** 1 **版**　2003 **年** 6 **月第** 2 **次印刷**
ISBN 7-81057-581-3/TP • 262
定价：26.00 **元**

内容简介

本书以关系数据库为主，从系统原理、应用开发及产品内核与标准三个角度，对数据库管理系统 DBMS 的基本概念与原理进行全新的多视角诠释。为使读者加深对其原理理论知识的理解，在各原理讲解之后，紧接着介绍该原理在实际商用 DBMS 中的具体体现，使读者不仅有理性认识，还有一定的感性认识。全书共分三篇，分别是原理及系统篇、应用设计篇和新技术、产品及标准篇。书中对目前相当流行的 SQL Server 的基本内核做了较全面详尽的介绍，可作为 SQL Server 或其他 RDBMS 开发者深入了解大中型 RDBMS 产品的快速入门指导书。

数据库的原理与设计通常是相互交叉、不易分清的，在介绍原理的同时，可能穿插有设计的内容，如不加以明确指出，容易让读者造成混乱。本教材将以通俗易懂的语言和图解的形式指出原理与设计之间的复杂关系。本书的主要特点是：将原理与应用设计内容紧密结合，展示一个实际应用系统的设计开发与实现。

本书内容丰富、结构新颖、系统性与实用性强，可作为计算机及相关专业专科生、本科生及研究生教材，也可作为从事数据库开发的技术人员的参考书。

前　言

本教材是为适应目前数据库技术迅猛发展的形势，以及为使数据库教学能够体现这些新发展目标而编写的。为此，对数据库的教学内容、结构做了进一步的调整、取舍和更新，力图使数据库这门重要的课程能够更快地反映当前数据库在应用开发、产品及标准等方面的新动向。

全书共分三篇，分别是：原理及系统篇、应用设计篇和新技术、产品及标准篇。

第一篇为原理及系统篇，由第 1 至第 7 章组成。第 1 章为数据库系统概述，本章一开始即从最原始的角度来分析、介绍数据库系统中的有关内容，由此形成将数据库原理、应用与设计这三部分融为一体的数据库总体结构图。建议读者一定要看看该内容，因为它首次明确指出了原理、应用及设计这三部分的联系及区别。了解了这一关系，能够为下面的学习梳理出一条清晰的主线，从而为学好数据库这门课程开一个好头。第 2 章介绍了在数据库的应用设计中用得非常广泛的一种概念数据模型，即：实体联系模型。第 3 章对目前占主导地位的关系模型，以及对其理论基础之一的关系代数和关系运算做了较全面的描述。第 4 章着重从实用角度，通过列举大量示例，对结构化查询语言 SQL（Structured Query Language）进行了比较详细的介绍。而第 5 章和第 6 章分别介绍了查询优化和数据库的存储结构。第 7 章是最能体现数据库功能的内容，主要从原理和应用的角度分别介绍了数据库的安全性、完整性、故障恢复和并发控制，是数据库课程的必学内容之一。

第二篇为应用设计篇，由第 8、第 9 和第 10 章组成。数据库原理学习完之后，所关心的问题就是如何将企业的管理数据存入数据库管理系统中，这正是第 8 章的目的。第 8 章将系统地介绍如何通过数据库的需求分析、概念设计、逻辑设计和物理设计等若干步骤一步一步地将企业的管理业务、数据等转变成数据库管理系统所能接受的形式，从而达到利用计算机管理信息的目的。第 9 章涉及的是关系模型的理论基础之一，该理论是指导数据库设计的重要依据，揭示了关系数据中最深沉的一些特性——函数依赖、多值依赖和连接依赖，以及由此引起的诸多问题，如冗余及更新问题、插入异常和删除异常等，通过理论引入，对关系模式的规范化进行了系统阐述。第 10 章将用一个实际的应用系统开发的实例，详细展示其中的精髓。读者只要遵从本章的设计、构建和开发的步骤，完全可以重现本章所设计系统，从而完成从理论到实践的跨跃。

第三篇为新技术、产品及标准篇，由第 11 至第 13 章组成。随着计算机技术以及相关学科的不断发展，以及各行各业对数据库应用需求的不断增长，使得数据库领域出现了许多令人惊异的新技术，这些新的数据库技术的出现与不断商用化，反过来又极大地推动了应用的发展。第 11 章将介绍目前数据库领域几种最新的技术及其进展。作为目前数据库市场颇有代表性的 RDBMS，SQL Server 得到了越来越多的应用。因此，第 12 章将其作为 RDBMS 产

品的代表，对 SQL Server 产品涉及的从原理、应用、设计到管理等各方面的知识进行了全面介绍。通过对该章的学习，可使读者对 DBMS 有更深的理解和更具体的感性认识。第 12 章的内容可以作为使用 SQL Server 产品的一个快速入门。第 13 章主要对 SQL-92 和 SQL：1999 这两个标准版本进行介绍。

本教材由陶宏才同志担任主编，主审为尹治本教授。

参加编写的人员及所编写部分为：陶宏才编写第 1 章的 1.1、1.2、1.3 和 1.4，第 3 章，第 7 章的 7.2、7.3 和 7.4，第 9 章，第 12 章，第 13 章的 13.1、13.3；楼新远编写第 11 章的 11.1、11.3；陈安龙编写第 2 章，第 4 章；苏金华编写第 5 章；袁连海和吴艾编写第 6 章；陈红梅编写第 7 章的 7.1，第 8 章；何虎编写第 10 章；桑应朋编写第 11 章的 11.2，第 13 章的 13.2。全书由陶宏才统纂、定稿，由尹治本教授主审。

本教材的编写与出版得到了西南交通大学计算机与通信工程学院领导和各位同仁、西南交通大学出版社的大力支持和帮助，作者表示衷心感谢。在此，作者还要特别感谢邓昌延教授以及作者的家属，感谢他（她）们对本书及作者的支持、理解和关心。

由于作者水平有限，书中难免会存在缺点和错误，敬请读者及同仁批评指正。

作　者

2001.7.9

目　录

第一篇　原理及系统篇

第三篇 新技术、产品及标准篇

第一篇

原理及系统篇

- **数据库系统概述**
- **实体联系数据模型**（Entity Relationship Model）
- **关系数据库系统** RDBS
- **结构化查询语言** SQL
- **查询优化**
- **数据库物理存储结构**
- **数据库的保护**

第 1 章

数据库系统概述

数据库的教材中，有关数据库的原理、应用与设计这三部分的内容常常相互交叉，不容易分清。本章将通过从最原始的观点来介绍数据库的有关内容，形成将此三部分融为一体的数据库总体结构图，以帮助读者既清晰它们之间的联系，又能明了它们之间的区别，从而为学好数据库这门课程开一个好头。

1.1 数据库管理系统及其总体概述

本节的主要目的是想在开始进入该课程学习之前，以最原始的观点出发来看待数据库系统中的若干基本问题，如数据库系统的出现、数据库管理系统应具备的功能、数据库的抽象层次、数据库应用系统的设计与开发、数据库语言等等，从而使读者能全面地了解数据库系统的总体结构，特别是数据库系统原理和数据库的应用与设计之间的密切联系，而这种联系也正是读者在学习数据库课程时最容易混淆和难以理解的。只有搞清了这种联系以及各自的内容框架，才能快速地学好数据库的原理，也为今后进行数据库的应用、设计和开发打下良好的基础。

1.1.1 基于文件系统的简单应用系统开发示例

要理解数据库管理系统的出现，应该从利用文件系统来开发管理应用软件和网络共享的观点来看待它。

假定我们要求利用 C/C++ 语言和基于操作系统的文件系统来开发一个简单的学生管理系统，且假定这个系统又由学生注册、学生选课、学生学籍和学生成绩等几个业务部分组成。

我们知道，C 语言有一个结构数据类型，它是用来存放相关数据的结构，利用这个结构类型，我们可以定义如下几个需要用到的结构，即：学生结构、课程结构、注册信息结构和成绩结构。

```
struct Student
{
    int     nStudNo;
    char    szStudName[20];
    char    cStudSex;
    int     nStudAge;
    char    szDept[30];
};
```

```
struct Course
{
    int     nCourseNo;
    char    szCourseName[20];
    char    szDept[30];
};
```

```
struct Enrolled
{
    int     nStudNo;
    int     nWhichTerm;
    char    cEnrolled;
    char    szMem[30];
};
```

```
struct Grade
{
    int     nStudNo;
    int     nCourseNo;
    int     nGrade;
};
```

这些数据结构的定义完全是为了存储学生管理中所要用到或产生的数据。结构一旦定义，即可根据要求编制一些基本的管理操作函数，以便录入、删改和统计数据。对于管理应用来说，最基本的操作实际上只有四个，即：增加（录入或插入数据）、删除、修改和查询，通常简称为“增删改查询”。其它业务功能基本上均是由这四个基本功能演变、组合而成，有些甚至只是名称上叫法不一样而已。因此，只要针对每个结构实现其上的“增删改”和查询功能，即可组合完成整个应用系统的管理功能。

为保证在下次进入管理软件时数据依然可用，还必须将以上的结构及其相应数据以文件系统中的文件形式存放在磁盘上。而文件的打开、读写和关闭等操作可利用操作系统的文件 I/O 操作功能。

根据以上思路即可完成一个简单的学生管理系统，这个应用系统的开发完成不是基于数据库管理系统，而是基于操作系统的文件系统。下面我们来看看这样一个利用文件系统来实现的应用系统有哪方面的不足或缺陷。

1.1.2　基于文件系统的应用系统缺陷

1. 大容量数据存储

前面所举例中，当文件中所存数据增长很大，如达到 500 GB 时，系统中没有如此大的内存来存放它，即使有这样大的内存，32 位计算机也不能直接寻址超过 4 GB 的数据。因此，要在示例系统中解决这个问题，只有按需要从磁盘或磁带上的数据文件中将相关的部分调入内存。为此，示例系统中就应增加处理这一情况的大量代码，而且每开发一个类似的管理应用系统，都要考虑这一情况的处理。

2. 多用户并发访问

上面示例系统只是供单用户使用的系统，但随着网络的发展和广泛应用，以及人们对数据共享的要求，使得应用系统应当面向多个用户同时访问和使用。但要供多用户访问，应用系统亦应做较大改动。因为单纯的文件系统并不支持多用户访问，要做到多用户访问就必须增加相应的处理。例如，当多个用户同时要对同一数据进行修改，应用系统就必须保证如何使得这种同时修改既有效又不致引起冲突。另一方面，应用系统还应让用户感觉不到是多个用户在同时使用同一个系统。

3. 数据一致性

从以上示例可以看出，同一数据可能会出现在多个结构中，即可能出现于多个数据文件

中，例如上例中的学生学号、课程编号。于是问题就出现了，即如何保证多个数据文件中同一数据的一致。例如，假定由于某种原因要修改学生的学号，那么由于该学生已选修有课程故其原来的学号在选课文件中亦存在，如果只修改学生文件中的学号而不修改选课文件中对应的学生学号，则会产生数据的不一致。文件系统本身并不支持或保证不同数据文件中同一数据的一致性，因此这一问题需要由应用系统来解决，即应用系统需要有相应的模块专门来处理这一问题。

另外，当用户正要修改某一数据而恰好此时系统崩溃，可能使得只有部分数据得到修改而其余部分未完成修改，于是也会造成数据的不一致。文件系统对此也是无能为力。

4. 安全性

我们知道，操作系统对数据文件的保护仅限于提供一个口令的安全机制，即任何一个用户只要知道了该口令即可突破这一安全屏障进入系统，从而可以看到整个数据文件的内容。这种安全机制对现实应用管理系统来说是绝对不想看到的，因为在现实业务管理中，并不希望无关的用户看到某些需要保密的数据。而在上述示例系统中，数据是存放在数据文件中的，要实现现实管理系统所想达到的安全性要求，示例系统也必须自行处理这一复杂的安全要求。

1.1.3 从文件系统缺陷来看数据库管理系统的出现及其基本功能

从以上文件系统缺陷的分析，可以看到为解决这些问题，设计的示例应用系统需要考虑做许多份外的但却不得不做的工作。从这些工作的要求看，它们又是开发每个管理应用系统所必须做的事情。因此，如果产生能够把这些工作抽取出来形成一个中间的、作为开发和应用的系统平台的这种思想，就促成了数据库管理系统 DBMS（Data Base Management System）的出现和发展。一旦有了这样一个开发和应用的系统平台，我们再设计前面的学生管理应用系统时，就可以不考虑或少考虑前面分析的一些系统要求，将这些系统要求交由数据库管理系统 DBMS 来做，而设计人员只专注于系统业务功能的实现，这样既可以大大加快系统开发的进度、减轻开发的难度，同时又可保证系统的可用性等方面的要求。

由上分析，一个数据库管理系统应该具备如下的基本功能：

1. 数据独立性

由前面的示例可以看出，由于数据结构在应用程序中定义，一旦想改变数据的结构则应用程序也必须做相应改动。也就是说，应用程序依赖于其所操作的数据结构，这样的缺点还会导致应用系统的升级较困难。因此，我们在抽出开发一个应用系统所必须的一些公共功能来作为数据库管理系统所应具备功能的同时，还应该考虑使应用程序与数据独立。只有做到数据的独立性，才能在以 DBMS 作为系统开发平台的应用系统研制时，使今后系统的修改和升级更加便利和快捷。

2. 并发控制

正如前面示例所谈到的，随着应用系统从面向单用户到面向多用户的转化，应用系统中

就应该考虑当多个用户同时访问同一数据对象时所发生的冲突。而作为应用系统开发和运行平台的 DBMS，应该对这种多用户的并发访问进行控制，以达到既允许多个用户对同一数据对象的同时访问，又能避免或较好地处理访问的冲突。使用户感觉不到自己是处于一个多用户并发访问的环境，而只觉得系统只有自己一人在使用一样。

3. 故障恢复

管理的数据对于企业来说是非常重要的，不允许在系统运行发生故障的情况下丢失任何数据。DBMS 应该保证应用系统即使发生故障，也能在故障清除之后恢复故障发生所丢失的数据。

4. 安全性

DBMS 应能保证存于其中的数据库的安全，使不具有访问权限的用户看不到其不该看到的数据，同时也应使具有权限的用户不能被拒绝访问其所应看到的数据。

5. 完整性

我们知道，现实系统中的数据本身不是孤立的，而是相互间具有一定联系或关系。作为管理这些数据的 DBMS 应该能够反映、保证数据的这种联系，且具有某种机制来维护数据的这种联系，以保证数据的完整、正确，这就是数据的完整性约束。有了这种完整性约束，DBMS 能够禁止破坏这种完整性的数据的录入，也能够禁止破坏完整性的数据改动，或者能对发生变化的数据进行一连串或级联式反应，以保证数据的一致性。

1.1.4　数据库管理系统的抽象层次

1. DBMS 中的抽象层次

DBMS 中的数据（Data）可在三个抽象层次上描述，如图 1-1 所示。每一级抽象有一个模式（Schema），分别是：概念模式（Conceptual Schema）、物理模式（Physical Schema）和外模式（External Schema）。

可用数据定义语言 DDL（Data Definition Language）来定义外模式和概念模式。关于 DDL，将在第 4 章的 SQL 语言部分介绍。所有的 DBMS 供应商也都提供有描述物理模式的 SQL 命令，不过，这些命令 SQL-92 标准不支持。三个模式的有关信息存储于数据字典中。

概念模式用 DBMS 的数据模型来描述存储的数据。对于关系 DBMS（Relational DBMS, RDBMS），它描述的是所有的关系（Relation），关系包括：实体及其联系，也就是说，它们均可用关系来描述。由于确定关系及其每个关系中的字段并非总是那么明确，为此就要进行概念数据库设计，以设计出一个好的概念模式。

物理模式用于指定存储细节，说明概念模式中描述的关系如何实际地存储于外存（如磁盘、磁带）中。其任务有：决定用什么文件组织来存储关系、创建索引来加速数据检索等。要完成此任务，就需要理解如何访问数据，进行物理数据库设计。

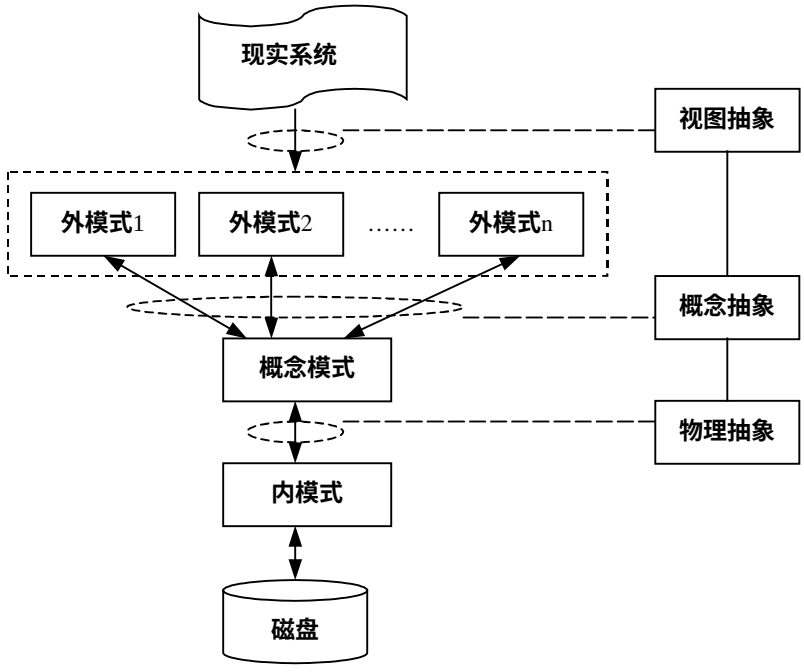

图 1-1 数据库管理系统抽象层次

外模式通常也是利用 DBMS 的数据模型来描述，允许对数据访问进行定制或授权，以满足不同用户（组）的需求。任何数据库只有一个概念模式和一个物理模式，因为它只有一组存储的关系。但它可有多个外模式，每个外模式针对的是一组特定的用户组。每个外模式由视图（View）和概念模式的关系组成。一个视图，概念上也是一个关系，但其记录并不存储于 DBMS 中。外模式的设计根据最终用户的需求进行。

在概念模式中不包含视图，因为视图可根据存储于概念模式中的关系计算得到。而且，如果要存储视图的话，会导致不必要的数据冗余，不仅浪费空间，而且容易引起数据的不一致。

对应三个模式分别形成三种级别的数据抽象（Data Abstraction），即：视图级抽象、概念级抽象和物理级抽象，从而为用户提供了一个数据的抽象视图，隐藏数据的存储结构和存取方法等细节。而数据抽象的工具即是利用一个或多个***数据模型*** DM（Data Model）。

视图级抽象是把现实世界抽象为数据库的外模式，概念级抽象再将数据库的外模式抽象为数据库的概念模式，而物理级抽象则是把数据库的概念模式抽象为数据库的内模式。

视图抽象将现实世界中的信息按照不同用户的观点抽象为多个逻辑数据结构。每个逻辑数据结构称为一个视图，描述了每个用户所关心的数据。每个视图抽象地描述了整个数据库的一个侧面。所有视图的集合形成了数据库的外模式。数据库外模式是面向用户的数据库模式。数据库系统中数据定义语言的视图定义机构提供了进行视图抽象的工具，可以用来定义视图的逻辑结构。

概念抽象把数据库的外模式抽象为数据库的概念模式。概念模式综合了外模式中所有的视图，反映了所有数据库用户所关心的现实世界的抽象，形成了数据库的整体逻辑结构。数据库系统中数据定义语言的逻辑数据库定义机构提供了进行概念抽象的工具，可以用来定义概念数据库模式的逻辑结构。

物理抽象将数据库的概念模式进一步抽象成为数据库的内模式。内模式抽象地描述了概念数据库如何在物理存储设备上存储。数据库的内模式包括两方面，一是存储策略的描述，包括数据和索引的存储方式、存储记录的描述、记录定位方法等；二是存取路径的描述，包括索引的定义、HASH 结构定义等。数据库系统中数据定义语言的物理数据库定义机构提供了进行物理抽象的工具，可以用来定义数据库的物理存储结构。

为更好地理解该抽象层次，我们可从以下两个方面来理解。

2. 从应用系统开发的角度来看待数据库的抽象层次

从图 1-1 可以看出，数据库的抽象层次在某种程度上反映的正是应用系统开发的过程。一般，开发应用系统的目的，最主要的一点是要把现实业务系统中的业务数据等存储到数据库管理系统管理的数据库中，即以数据库的内模式形式存放，这个内模式是数据库管理系统已经实现了的物理数据模型。

从 DBMS 本身来看，每一种 DBMS 的实现，均是建立在某一种数据模型基础之上的，例如：关系型 DBMS 即是建立在关系数据模型基础之上，像 Sybase、Oracle、Informix、DB2、Microsoft SQL Server 数据库管理系统均是关系型 DBMS，它们是通过关系来存储数据以及数据之间的联系；而层次型 DBMS 和网状型 DBMS 则分别建立在层次数据模型和网状数据模型基础之上。DBMS 所支持的数据模型与 DBMS 之间的关系，类似于文件系统与操作系统之间的关系，操作系统存取数据是通过文件系统来实现的，而 DBMS 存取数据则是通过数据模型这样的数据结构来实现。

从数据库应用系统的设计与开发角度看，要设计开发一个数据库管理应用软件，需要将现实世界业务管理系统中所涉及的所有数据及其数据之间的联系，通过数据抽象，利用数据模型来表达、描述，逐渐演变、转化成实际 DBMS 支持的（即：可以实现的）数据模型，对于 RDBMS，即是关系数据模型。

目前数据库系统的三级抽象的原因在于：

（1）由于人类认识事物是一个从片面到全面，从个体到整体的过程。

（2）目前开发的管理应用系统不再只是单个业务系统，而是一个综合的业务系统，即所设计与开发的系统不再只是为单个业务系统的用户服务，而是要为整个系统中所有业务系统的用户服务。作为综合业务，其中可能有很多数据在各个部门业务中都会出现，即存在重复的、相同的数据，这就要求开发人员在设计综合业务系统时，要详细调查各个部门所关心的数据、联系及其业务，然后综合之，消除重复的部分并优化，得到一个综合的可为系统内各个业务部门服务的数据模型，称之为概念模式（模式是数据模型描述的结果，是其具体的体现）。各个部门只看到该模型中他们所关心的那一部分，称之为该部门的用户视图，亦称外模式。因此，概念模式只有一个，而外模式则可有多个。

（3）目前数据模型存在的缺陷。作为一个较完善的数据模型，应该具有三个基本的特点：①能够准确地描述、表达现实系统中的数据及其联系；②能够容易为普通的用户所理解；③能够容易被计算机所实现。不幸的是，RDBMS 所支持的关系数据模型，能够满足第一和第三个要求，对第二个要求却差强人意。因此，人们不得不寻找一些中间的数据模型，来帮助进行管理系统的开发，这样的模型有：实体联系模型（Entity Relationship，简称 ER 模型，）、面向对象

的数据模型等。这些模型的使用，又带来了另外的问题，即：由于它们不是 DBMS 所支持的数据模型，因此用这些数据模型描述的模型还需要经过转化，演变成 DBMS 所支持的数据模型。

当前 OO（Object-Oriented，面向对象）模型很流行，也开发出支持 OO 模型的 DBMS，即 OODBMS，但还未商业化。不过，OO 模型也许将会成为第一个满足数据模型三个特点的理想模型。这样，整个开发与设计过程中，只用一种数据模型即可，不用进行任何与其它模型的转换。

3. 从数据库的抽象层次来进行数据库应用管理系统的设计与开发及其设计工具

数据库的抽象层次反过来对应用系统的设计与开发可以起到指导作用。实际上，从前面的介绍也可以看出这一点，即：可以将应用系统的设计与开发按自顶向下或自底向上的方法进行。所谓自顶向下，即是先总体后个体，从抽象层次上看，是先概念抽象然后视图抽象，具体来说，先从整个应用系统的总体要求出发，构造总体的应用系统逻辑模型，然后再根据各个业务机构的业务要求，从总体逻辑模型出发构造各业务子系统的逻辑模型，形成其各自的视图。而自底向上则是先个体后总体，从抽象层次上看，是先视图抽象然后概念抽象，具体方法即是：先从各业务子系统的业务要求出发，构造各业务子系统的逻辑模型，然后在此基础上综合、优化，形成整个应用系统的总体逻辑模型。

至于设计工具，在此主要是指所用的数据模型，从前面介绍看，在视图抽象和概念抽象阶段，主要使用抽象级别较高的数据模型——ER 模型，在从概念抽象到物理抽象的过程当中，还应将 ER 模型转换成具体的 DBMS 所支持的数据模型，对 RDBMS 来说，即是关系数据模型，最后由 RDBMS 将关系数据模型映射到相应的物理数据模型。

综上所述，数据库的抽象层次既体现了数据抽象的分层性，也将应用系统设计开发的分步性巧妙地结合在一起。使得抽象层次与应用系统的开发方法相互补充、相互说明，也使数据库的原理和数据库的应用与设计达到完美的结合。

1.1.5 数据独立性（Data Independence）

使用 DBMS 所获得的一个非常重要的优点是数据的独立性，也就是说，应用程序不会因数据结构和存储的变化而变化。数据独立性是通过使用数据抽象的三个层次取得的，特别是概念模式和外模式。数据独立性分逻辑数据独立性和物理数据独立性两种。

逻辑数据独立性（Logical Data Independence）是指，当基层数据重新组织，概念模式发生变化时，导致视图定义也要作相应改变，以便能得到与以前相同的关系。在此，由于视图关系没有被实际存储，改变其定义很容易实现，因此，具有较好的数据独立性。于是，外模式能使用户免受因逻辑结构（即概念模式）的变化而引起的变化，这种特性即为逻辑数据独立性。

而物理数据独立性（Physical Data Independence），则能够使用户远离数据物理存储的变化。概念模式可隐藏具体细节，如：数据如何分布于磁盘上、文件结构如何、索引如何选择等等。只要概念模式不变，可以改变这些存储细节，而勿需改变应用。当然，性能可能会受到一定影响。

1.1.6　数据库语言与 ODBC

很多人已经知道，目前最流行的数据库语言是 SQL 语言（Structured Query Language，即结构化查询语言，简称 SQL 查询语言）。该语言不同于其它的编程语言，如 C/C++、Pascal 等等，它分为两个子语言，即：数据定义子语言 DDL（Data Definition Language）和数据操纵子语言 DML（Data Manipulation Language）。SQL 将数据的定义与操纵分开进行，这与高级编程语言中将数据的定义与操作放在一起不同。

数据定义子语言用来创建、修改和删除要存放到数据库中的数据结构（如表、视图等）、数据语义（如各种完整性约束或限制）及其它数据库对象（如用户定义数据类型、存储过程、触发器、游标等）。从前面的介绍可知，数据库中所用的数据结构等即是在此时用 SQL 语言创建定义的。由于 SQL 是基于关系模型的，所以它只能将分析设计所得的关系模型，利用适当的数据定义语句，存放到数据库中。数据一旦定义，即由数据库存放于数据字典 DD（Data Dictionary）中，包括数据的存储路径、类型、长度、安全权限等等信息。

SQL 语言最方便之处在于数据的操纵，是通过数据操纵子语言来进行的。其方便性在于，操纵数据（如增删改）时，勿需再指出数据的存储路径、类型、长度等细节，而只需通过名字即可实施对数据的操纵。细节的处理交由数据库管理系统执行，而勿需用户操心，这样可大大方便用户对数据库的操作。

注意：DDL 与 DML 中都有修改和删除命令，但两者的关键词是不同的。一般来说，在 DDL 中，修改用 Alter，删除用 Drop；而在 DML 中，修改用 Update，删除用 Delete。

SQL 语言既适合控制台上交互式的操作，也可嵌入到高级编程语言中使用。由于关系数据库管理系统产品众多，且其支持的 SQL 语言在功能、语法上存在一定差异，为便于应用程序的移植和互操作，应用程序一般是通过 ODBC（Open DataBase Connectivity，开放数据库连接）、JDBC（Java DataBase Connectivity）或其它应用编程接口 API（Application Programming Interface）来完成对数据库的操纵。这些接口可连接至不同的数据源（即不同的数据库），提供一致的高级调用接口，将 SQL 命令语句（如有必要的话，这些 SQL 语句可能会做相应转换，以符合指定的 DBMS 所支持的 SQL 语法）提交给对应的数据源，并返回查询结果，从而实现对所连数据库的操纵。正是由于 ODBC 这类标准接口的存在，目前大多数高级编程语言或其它专门的数据库开发语言，如 C/C++、Delphi 等，才得以实现对数据库中数据的操纵。

1.1.7　数据库系统总体结构

综上所述，我们可以把数据库原理、数据库应用及数据库设计这三个部分通过图 1-2 有机地联系起来，形成数据库系统的总体结构图，实际上应该是一个数据库的总体印象图，主要是为了展示数据库的这三个部分之间内在的紧密联系。通过该图，可以清楚地看出它们之间既相互独立又相互关联的关系。明白了它们之间的关系，对学好数据库这门课程有极大的帮助。